МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
"ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"

МЕТОДИЧНІ ВКАЗІВКИ

до виконання індивідуального завдання
з навчальної дисципліни

«Програмування автоматизованих технічних комплексів»

для студентів денної та заочної форми навчання за спеціальністю «Прикладна механіка», освітня програма «Моделювання технічних систем»

Затверджено редакційно-
видавничою радою університету,
протокол № 2 від 27.06.2025 р.

Харків
НТУ «ХПІ»
2025

Методичні вказівки до виконання індивідуального завдання з навчальної дисципліни «Програмування автоматизованих технічних комплексів» для студентів денної та заочної форми навчання за спеціальністю «Прикладна механіка», освітня програма «Моделювання технічних систем» / уклад.: В. В. Клітной, М. Г. Стрижак. – Харків : НТУ «ХПІ». – 2025. – 49 с.

Укладачі: В. В. Клітной
М. Г. Стрижак

Рецензент А. С. Роговий

Кафедра «Деталі машин та гідропневмосистеми»

ВСТУП

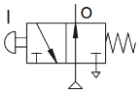
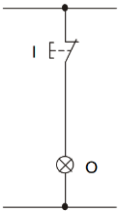
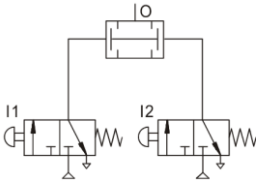
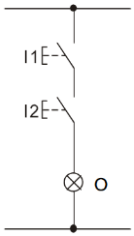
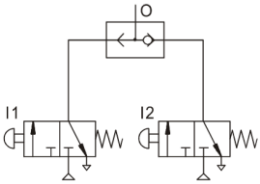
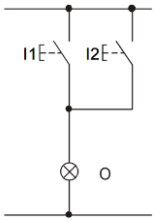
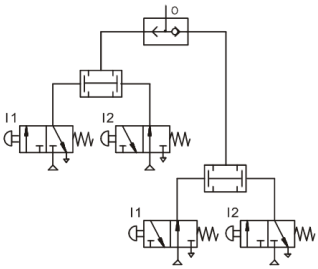
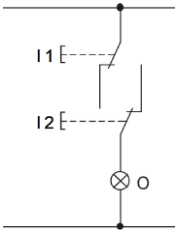
В багатьох сучасних технологічних системах застосовуються автоматизовані технічні системи як засоби реалізації різноманітних виробничих та логістичних процесів. Їх область застосування не обмежується лише промисловим виробництвом, а розповсюджується також на сфери логістики, торгівлі та багато інших.

Сучасний розвиток технічних систем та автоматизованих комплексів вимагає від фахівця знань і практичних навичок у сфері програмування. Навчальна дисципліна «Програмування автоматизованих технічних комплексів» спрямована на формування у студентів компетентностей, необхідних для розробки, аналізу та оптимізації програмного забезпечення, що використовується в автоматизованих технічних пневмосистемах.

Ці методичні вказівки призначені для студентів спеціальності G9 – Прикладна механіка. Вони містять теоретичні відомості щодо щодо етапів створення проекту автоматизації технологічного процесу, перевірки його працездатності та випробування на лабораторному стенді.Наявний приклад виконання індивідуального завдання і вихідні дані.

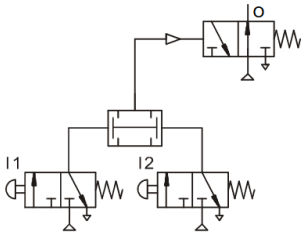
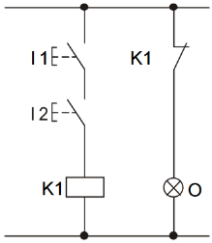
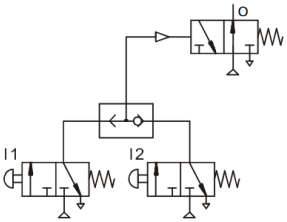
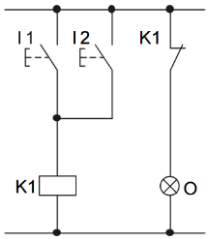
УМОВНІ ПОЗНАЧЕННЯ

Таблиця 1 – Умовні позначення елементів логічних зв'язків у системах

Назва елемента у середовищі Fluidsim	Логічний символ	Реалізація на базі пневмоелементів	Електрична схема
1	2	3	4
NOT			
AND			
OR			
XOR			

Таблиця 1 – продовження

Назва елемента у	Логічний символ	Реалізація на базі пневмоелементів	Електрична схема

середовищі Fluidsim			
1	2	3	4
NAND			
NOR			

1. ПРОГРАМУВАННЯ ДЛЯ ТЕХНОЛОГІЙ АВТОМАТИЗАЦІЇ

Автоматизацію можна визначити як "*використання процесорів для автоматичного виконання технічних процесів*".

Відправною точкою для автоматизації є технічний процес, який повинен бути виконаний таким чином, щоб досягти певних цілей (наприклад, ефективність, безпека, доступність).

Для того, щоб це відбувалося автоматично, тобто без постійного втручання людини, необхідне обладнання автоматизації. Воно реалізує функції реєстрації, обробки та виведення сигналів.

Реалізація алгоритму обробки відбувається у вигляді логічних, арифметичних зв'язків (рис. 1.1). У минулому виконання таких функцій базувалося на різних видах допоміжної енергії (наприклад, електричної, пневматичної, гідравлічної); зараз процеси керування технологічними операціями реалізуються майже виключно за допомогою *комп'ютерних програм*.

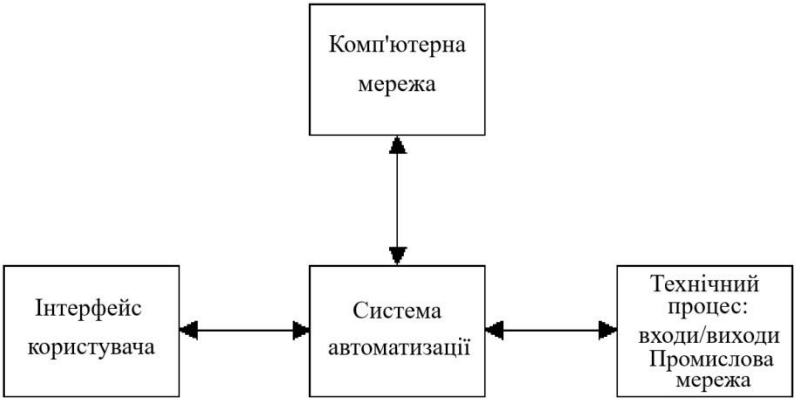


Рисунок 1.1 – Схема реалізації системи автоматизації технологічного процесу

Алгоритми обробки сигналів, розглянуті у даних методичних вказівках, розробляються, тестуються та оптимізуються за допомогою програми FST2.

Мета методичних вказівок полягає у вивченні і випробуванні основних, необхідних функції програмного забезпечення FST. Всі приклади були розроблені з використанням програми FST версії 4.10.

1.1. Програмування для техніки автоматизації або ПК

На відміну від програм, написаних, наприклад, для ПК або міні-ЕОМ, програми для технологій автоматизації повинні функціонувати "без постійного втручання людини". Наприклад, в той час як текстовий процесор, в який вводиться цей текст, вимагає постійної присутності людини, зварювальний робот повинен зварювати кузов автомобіля без постійного втручання людини. Однак для того, щоб "технічний процес" відбувся, людина, природно, необхідна в першу чергу. Таким чином, існує потреба в плануванні та програмуванні, а також у втручанні користувача.

Планування та програмування здійснюється на комп'ютерах за допомогою програмного забезпечення, розробленого спеціально для цієї мети. Далі програми виконуються в контролері.

Це являє собою значну різницю між програмуванням для типових додатків для ПК і для типових додатків для автоматизації.

У технологіях автоматизації комп'ютери для програмування та комп'ютери-контролери є двома різними, незалежними блоками (рис. 1.2).

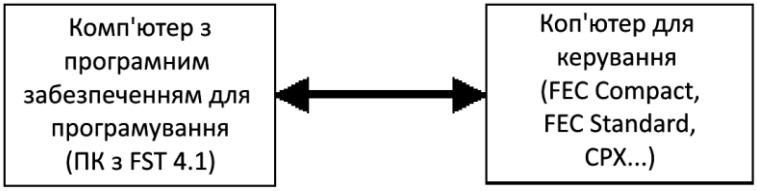


Рисунок 1.2 – Схема реалізації процесу автоматизації технологічного процесу

Таким чином, у технологіях автоматизації ми завжди повинні припускати, що необхідно щонайменше дві машини: машина для програмування і машина-контролер, або просто: контролер.

1.2. Основні правила створення проекту у FST

У табл. 1.1 детально розглянуті процедури програмування для середовища FST, що є загальною основою для розробки проєктів автоматизації з базовими завданнями.

Таблиця 1.1 – Базові процедури програмування для середовища FST

№ п.п.	Задача	Коментар
1	Створити новий проєкт	У меню Project
2	Назвати проєкт	У діалоговому вікні
3	Обрати відповідний контролер, ввести коментар до проєкту	Можна використовувати програмне забезпечення FST для програмування різних типів контролерів
4	Конфігурація 1/0 (вводу/виводу)	Кожен проєкт автоматизації має вхідні та вихідних дані
5	Програмування	Наявність базової програми (Program 0) є обов'язковою. Прийняття рішення про тип мови програмування: структурна схема (Ladder Diagram) або список операторів (Statement list)
6	Створити проєкт	Система пропонує автоматично
7	Завантажити проєкт	Для цього має бути встановлено з'єднання з контролером
8	Перевірка	Надає можливість виправити, покращити, оптимізувати програму
9	Документ	Опис, коментарі, можливість роздрукувати

2. СТВОРЕННЯ FST–ПРОЄКТУ КЕРУВАННЯ ГАРАЖНИМИ ВОРОТАМИ

Мета роботи – набуття практичних навичок у створенні програми керування роботою гаражних воріт, ознайомлення з загальними принципами складання і моделювання роботи схем

пневмоприводів з контролерним керуванням з використанням програми FluidSim.

Завдання: гаражні ворота повинні відчинятися та зачинятися зсередини та ззовні.

Постановка задачі: гаражні ворота повинні управлятися таким чином, щоб:

– двері можна закрити в будь-який момент зсередини та ззовні;

– двері можна відчинити ззовні лише при одночасному натисканні на ключовий вимикач і кнопку OPEN;

– двері можна відчинити зсередини в будь-який момент;

– при відкриванні двері завжди рухаються до кінцевого вимикача (вгору);

– двері зачиняються лише тоді, коли натиснута кнопка CLOSE.

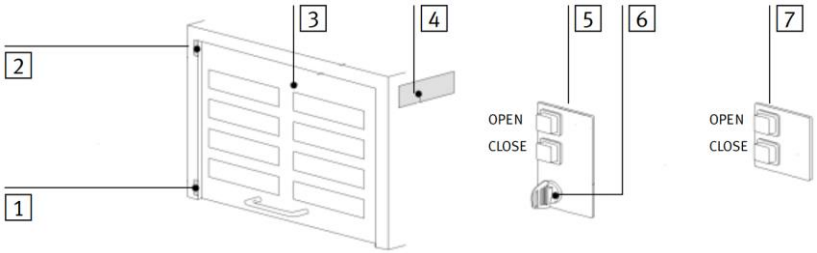


Рисунок 2.1 – Схема гаражних воріт:
1 – кінцевий вимикач (нижній); 2 – кінцевий вимикач (верхній); 3 – гаражні ворота; 4 – двигун підйому/опускання; 5 – зовнішня частина контролера гаражних воріт; 6 – ключовий вимикач; 7 – внутрішня частина контролера гаражних воріт

Алгоритм програми керування роботою гаражних дверей наведений на рис. 2.2. В якості контролера використовується IPC FEC FC20, тобто компактний контролер з 12 входами і 8 виходами (будь-який інший контролер FST також можна застосовувати).

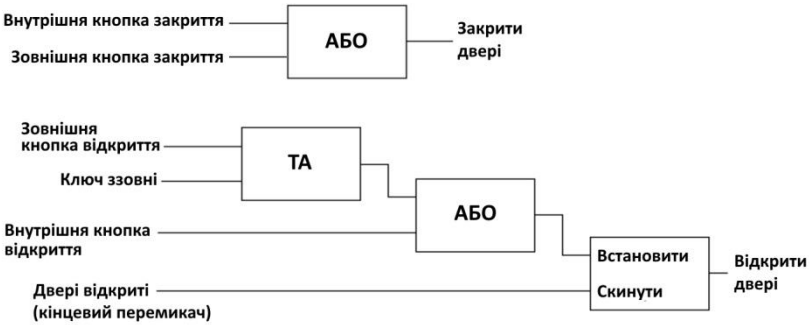


Рисунок 2.2 – Алгоритм програми керування роботою гаражних дверей, реалізований на базі контролера IPC FEC FC20

2.1. Проєкт, входи/виходи, програма, IF...TO...OTHRW

2.1.1 Створення нового проєкту

Створимо новий проект за методикою, наведеною у табл. 1.1.

%0.1 Проєкт, входи/виходи, програма, IF ... TO ... OTHRW

%0.1.1 Проєкт

– Необхідно створити новий проєкт (рис. 2.3).

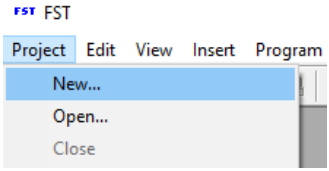


Рисунок 2.3 – Діалогове меню створення нового проєкту

- Необхідно присвоїти проєкту ім'я «GARAGE» (рис. 2.4) і підтвердити вибір натисканням кнопки «OK».

Рисунок 2.4 – Вікно присвоювання ім'я проєкту

%0.1.2 Вибір типу контролера

– Необхідно обрати відповідний контролер (на рис. 2.5 – FEC Compact) і (за бажанням) додати коментар.

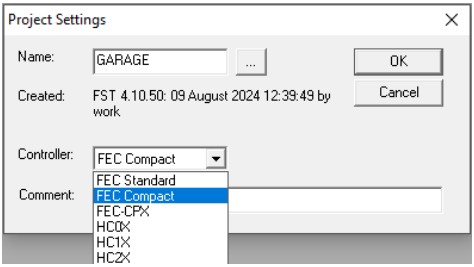


Рисунок 2.5 – Вікно вибору типу контролера

%0.1.3 Конфігурація 1/0 (вводу/виводу)

Нобхідно відкрити конфігурацію вводу/виводу, двічі клацнувши запис у вікні проекту (рис. 2.6).

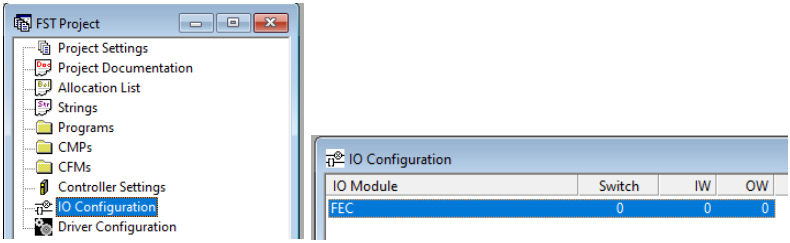


Рисунок 2.6 – Вікно конфігурації вводу/виводу

З рис. 2.6 видно, що модуль вводу/виводу повинен мати певні присвоєні значення: у стовпці IW вводяться входи, які починаються з I0.0, а у стопці OW – виходи, що починаються з O0.0.

2.1.2 Конфігурація вводу/виводу для стандарту FEC

Для застосування модулю вводу/виводу:

– натискаємо правою кнопкою миші на порожній конфігурації вводу/виводу і обираємо «Вставити» з контекстного меню або двічі натискаємо на порожній конфігурації;

– відкриваємо меню і обираємо карту вводу/виводу, яка потрібна для створюваної програми (рис. 2.7).

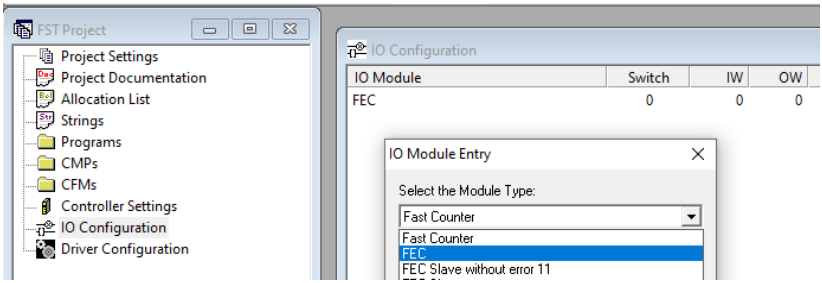


Рисунок 2.7 – Вікно карти модулю вводу/виводу

FEC-FC20-FST (рис. 2.8) має 12 входів і 8 виходів. Входи згруповані в 2 групи з одним байтом у кожних 8 входах (8 біт, 8 входів).

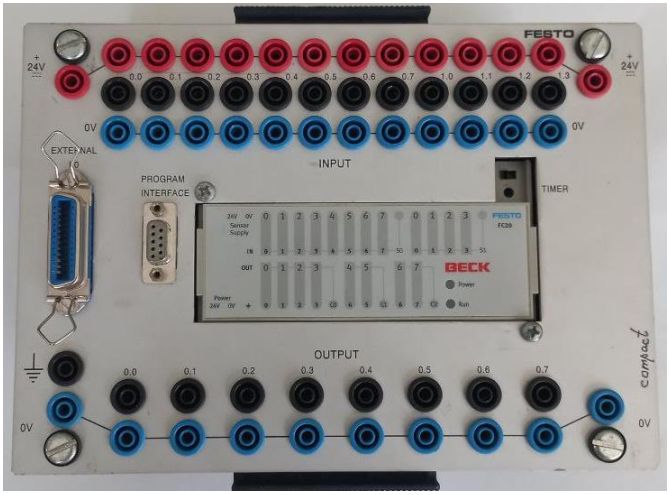


Рисунок 2.8 – Контролер FEC-FC20-FST

Виберіть FEC з меню (рис. 2.7), і вхідне слово буде підписано для кожного байта. Таким чином, наступний байт буде розподілено автоматично:

I0.0 ... I0.7

I1.0 ... I1.3

(O.00) O0.7

2.1.3 Адреси вводу/виводу

На прикладі керування роботою гаражних воріт видно, що для функціонування системи необхідні датчики, виконавчі механізми та контролер.

Датчики та виконавчі механізми є незалежними пристроями (кінцевий вимикач, перемикач, датчик температури, клапан, контакт, двигун...). Щоб контролер міг використовувати ці пристрої, між датчиками/приводами та контролером повинен бути встановлений зв'язок (рис. 2.9).

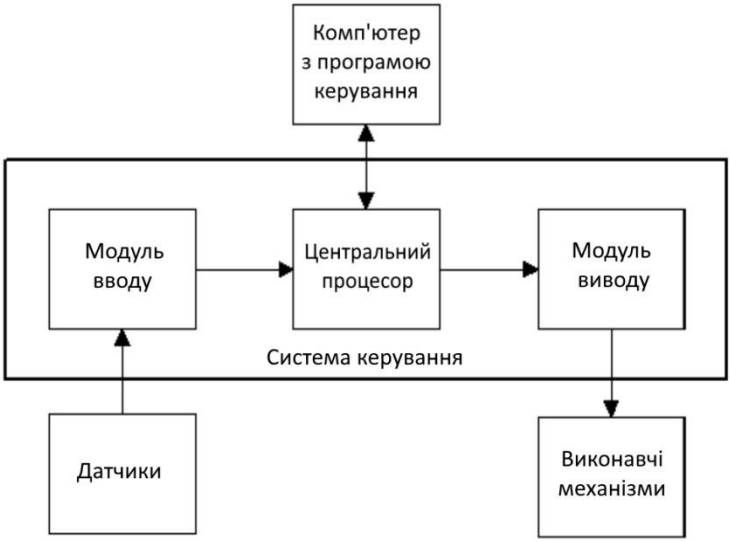


Рисунок 2.9 – Функціональна схема системи керування технологічним процесом

Підєднання датчиків до виконавчих механізмів відбувається через модулі вводу/виводу. Ці модулі вводу/виводу потребують імені, щоб центральний процесор міг звертатися до них. У FST ім'я завжди є словом, тобто «IW» для «вхідного слова» і «OW» для «вихідного слова». У межах «слова» можна отримати доступ до одного біта.

Великі або вкладені контролери можуть мати багато модулів вводу/виводу.

По суті, це не що інше, як "адреса". Якщо, наприклад, необхідно розподілити дані на 16 різних входів, то контролер розподілить дані для вихідного слова OW66 (рис. 2.10).

Дані розподіляються наступним чином:

O66.0, O66.1, O66.2 … O66.15

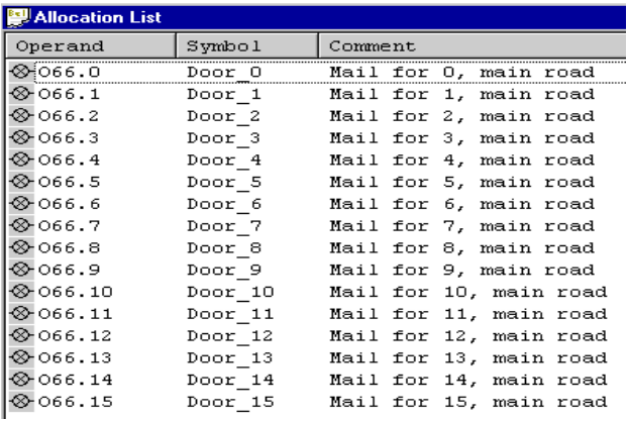


Рисунок 2.10 – Список операторів (Allocation List)

2.1.4 Програмування

На практиці програмування здебільшого починається з внесення вхідних та вихідних даних до списку операторів. Список операторів відкривається подвійним клацанням миші (рис. 2.11).

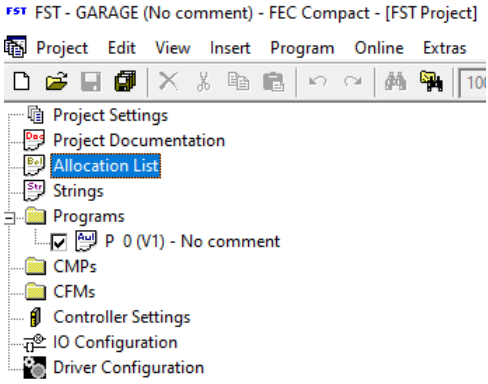


Рисунок 2.11 – Відкриття списку операторів (Allocation List)

У порожньому списку операторів можна ввести перший оператор ще одним подвійним натисканням лівої кнопки миші або клацнувши правою кнопкою миші, щоб викликати контекстне меню (рис. 2.12).

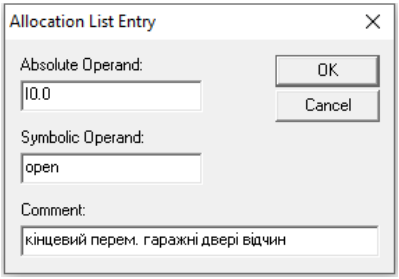


Рисунок 2.12 – Заповнення даних списку операторів

Тепер вводять усі оператори до списку. На рис. 2.13 наведено можливий список операторів (Allocation List), що відповідає схемам на рис. 2.1 та 2.2.

Allocation List

Operand	Symbol	Comment
O0.0	Relay_op	K1: реле відчиняє двері
O0.1	Relay_clo	K2: реле зачиняє двері
I0.0	open	кінцевий перем. гаражні двері відчин
I0.1	closed	кінцевий перем. гаражні двері зачин
I0.2	Open_in	кнопковий перемикач всередині Відкри
I0.3	Close_in	кнопковий перемикач всередині Закрит
I0.4	Open_out	кнопковий перемикач ззовні Відкрити
I0.5	Close_out	Кнопковий перемикач ззовні Закрити
I0.6	Key	Ключ ззовні

Рисунок 2.13 – Приклад списку операторів із внесеними даними

Правила створення списку розподілу:

– Адреса оператора повинна відповідати запису у конфігурації вводу/виводу. Якщо у конфігурації вводу/виводу як вхідне слово було введено IW20, то тут слід використовувати адресу I20.0 і т.д.

– Символ може містити до 9 символів, без спеціальних символів, а також без наперед визначених символів.

Наприклад, I0 та V10 заборонені як символьні операнди, оскільки I0 використовується для входу, а V10 – для константи.

– Коментар може складатися з будь-яких символів.

Після створення списку розподілу необхідно написати програму. Кожен проект FST вимагає програми з номером 0.

Клік правою кнопкою миші на директорії «Programs» («Програми») на панелі інструментів проекту відкриває контекстне меню з можливістю вставки нової програми. Або скористайтеся піктограмою «Нова програма» («New Program») чи меню «Вставити» («Insert»).

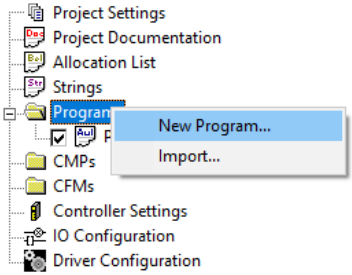


Рисунок 2.14 – Створення нової програми

Далі необхідно обрати потрібну мову програмування. Зверніть увагу, що не можна автоматично переключатися між списком операторів і структурною схемою. Таким чином, вибір однієї з двох мов застосовується до всієї програми (рис. 2.15). З іншого боку, програми на різних мовах програмування можуть бути змішані у проєкті – програма 0 може бути написана у вигляді списку операторів (STL), тоді як програма 12 – у вигляді структурної схеми (LDR).

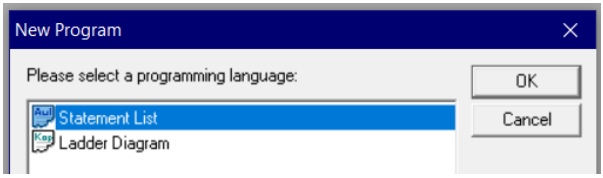


Рисунок 2.15 – Вибір типу програми

Далі необхідно обрати наступні параметри (рис. 2.16):

Type: Program або CMP або CFM (у даному прикладі залишаємо на рівні програми)

Number: 0 ... 63 (у даному прикладі залишаємо 0)

Version: 1 ... 9 (у даному прикладі залишаємо 1)
Comment: Це програма 0 керування гаражними воротами

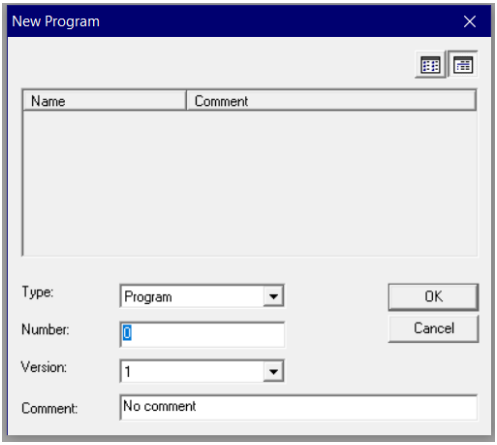


Рисунок 2.16 – Вибір параметрів програми

Натискання кнопки OK відкриє вікно програми, яке поки що є порожнім (рис. 2.17). Якщо ярлик не з'явився – можна відкрити його, натиснувши на пункт «Shortcuts» («Ярлики») в меню «View» («Вид»).

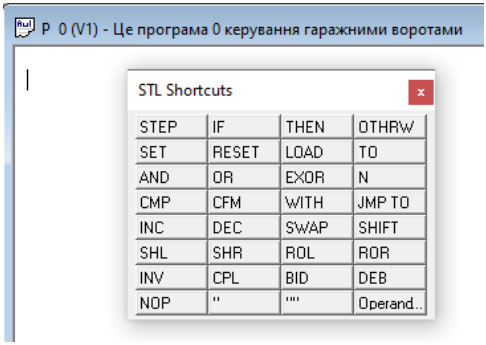


Рисунок 2.17 – Вікно написання програми

Далі необхідно описати функціонування гаражних воріт з контролерним керуванням.

Двері повинні відчинятися, якщо (рис. 2.2):

– кнопка натиснута всередині **АБО** (**OR**)

– кнопка натискається ззовні **І** (**AND**) одночасно спрацьовує ключовий вимикач.

У списку операторів FST можна використовувати майже ті самі слова (рис. 2.18).

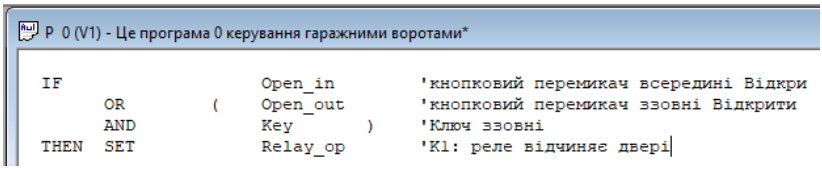


Рисунок 2.18 – Текст програми керування гаражними воротами

При цьому майже всі записи можна робити за допомогою комбінацій клавіш, миші або набору тексту на клавіатурі (або комбінації будь-якого з цих способів).

Можна використовувати символи замість абсолютної адреси оператора. Тому наступна програма є еквівалентною:

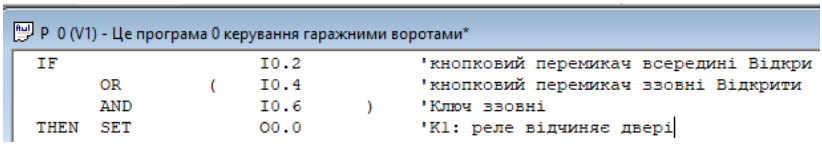


Рисунок 2.19 – Текст програми керування гаражними воротами

За допомогою рядка THEN SET O0.0 двигун був увімкнений для відкриття дверей, але він не був вимкнений. Однак коли двері відчинилися, двигун повинен бути вимкнений (рис. 2.20).

```
IF                open            'кінцевий перем. гаражні двері відчин
THEN  RESET       Relay_op        'K1: реле відчиняє двері
```

Рисунок 2.20 – Текст програми, що відповідає за вимкнення двигуна привода відкриття дверей

При закритті дверей відповідна кнопка повинна залишатися натиснутою безперервно з міркувань безпеки.

Тому ми можемо сформулювати це наступним чином: двері зачиняються, **ЯКЩО** (**IF**) натиснута кнопка CLOSE з внутрішньої

АБО (**OR**) зовнішньої сторони **І** (**AND**) двері не зачинені. **ІНАКШЕ** (**OTHERWISE**) вони залишаються на місці.

```
IF              (    Close_in       'кнопковий перемикач всередині Закрит
      OR             Close_out )    'Кнопковий перемикач ззовні Закрити
      AND     N      closed         'кінцевий перем. гаражні двері зачин
THEN  SET            Relay_clo      'K2: реле зачиняє двері
OTHRW RESET          Relay_clo      'K2: реле зачиняє двері
```

Рисунок 2.21 – Текст програми, що відповідає безпечному функціонуванню дверей

Також необхідно додати загальні умови:

– двигун не може працювати в обох напрямках одночасно. Отже, для обох напрямків необхідно задавати одночасно і протилежний напрямок також.

– двигун залишається на місці при одночасному натисканні кнопок CLOSE і OPEN.

З урахуванням вище зазначеного, фінальний текст програми наведений на рис. 2.22.

```
IF                   Open_in        'кнопковий перемикач всередині Відкри
      OR        (    Open_out       'кнопковий перемикач ззовні Відкрити
      AND            Key       )    'Ключ ззовні
      AND     N      Relay_clo      'K2: реле зачиняє двері
      AND     N      Close_in       'кнопковий перемикач всередині Закрит
      AND     N      Close_out      'Кнопковий перемикач ззовні Закрити
      AND     N      open           'кінцевий перем. гаражні двері відчин
THEN  SET            Relay_op       'K1: реле відчиняє двері
IF                   open           'кінцевий перем. гаражні двері відчин
      OR             Close_in       'кнопковий перемикач всередині Закрит
      OR             Close_out      'Кнопковий перемикач ззовні Закрити
THEN  RESET          Relay_op       'K1: реле відчиняє двері
IF              (    Close_in       'кнопковий перемикач всередині Закрит
      OR             Close_out )    'Кнопковий перемикач ззовні Закрити
      AND     N      closed         'кінцевий перем. гаражні двері зачин
      AND     N      Relay_op       'K1: реле відчиняє двері
      AND     N      Open_in        'кнопковий перемикач всередині Відкри
      AND     N      Open_out       'кнопковий перемикач ззовні Відкрити
THEN  SET            Relay_clo      'K2: реле зачиняє двері
OTHRW RESET          Relay_clo      'K2: реле зачиняє двері
```

Рисунок 2.22 – Програма керування роботою гаражних дверей

Однак при створенні програм необхідно дотримуватися умови їх доступності для інших користувачів, яку забезпечує наявність коментарів (рис. 2.23).

```
"" Project: GARAGE
"" Author: Bilous A.N.
"" Date: 01.12.2024"

""Open garage dor
 IF                  Open_in        'кнопковий перемикач всередині Відкри
       OR        (   Open_out       'кнопковий перемикач ззовні Відкрити
       AND           Key       )    'Ключ ззовні
       AND     N     Relay_clo      'K2: реле зачиняє двері
       AND     N     Close_in       'кнопковий перемикач всередині Закрит
       AND     N     Close_out      'Кнопковий перемикач ззовні Закрити
       AND     N     open           'кінцевий перем. гаражні двері відчин
 THEN  SET           Relay_op       'K1: реле відчиняє двері
 IF                  open           'кінцевий перем. гаражні двері відчин
       OR            Close_in       'кнопковий перемикач всередині Закрит
       OR            Close_out      'Кнопковий перемикач ззовні Закрити
 THEN  RESET         Relay_op       'K1: реле відчиняє двері
 IF              (   Close_in       'кнопковий перемикач всередині Закрит
       OR            Close_out )    'Кнопковий перемикач ззовні Закрити
       AND     N     closed         'кінцевий перем. гаражні двері зачин
       AND     N     Relay_op       'K1: реле відчиняє двері
       AND     N     Open_in        'кнопковий перемикач всередині Відкри
       AND     N     Open_out       'кнопковий перемикач ззовні Відкрити
 THEN  SET           Relay_clo      'K2: реле зачиняє двері
 OTHRW RESET         Relay_clo      'K2: реле зачиняє двері
```

Рисунок 2.22 – Програма керування роботою гаражних дверей (з коментарями)

Зазначимо правила заповнення списку операторів (Allocation List):

– список операторів можна редагувати до натискання клавіші «Enter». Після натискання програма буде відформатована, а коментарі будуть додані до списку операторів;

– між кожною операцією та кожним оператором має бути принаймні один пробіл;

– в іменах, операторах та операціях не допускаються пробіли;

– при введенні інформації, яку FST не розпізнає, вважається, що це визначення нового оператора. Програма надає запит, наведений на рис. 2.23. FST припускає, що «Nonsens» є символом нового оператора. Просто натисніть кнопку «Cansel» (або клавішу «Esc») і зробіть виправлення;

– принцип роботи FST грунтується на наборі булевих множин.

Булева множина складається з **IF** і **THEN**, і, можливо, може додатково містити **OTHRW**. **IF**; **THEN** та **OTHRW** можуть зустрічатися у множині лише один раз.

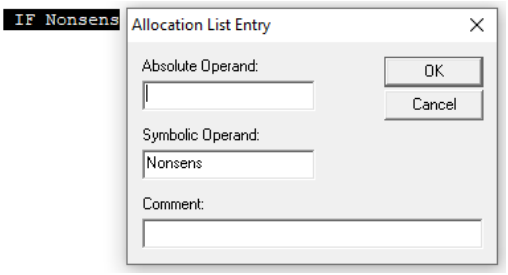


Рисунок 2.23 – Запит у середовищі FST на введення нового оператора

2.1.5 Програма у вигляді структурної схеми

Після того, як список операторів (Allocation List) створено, переходимо до написання програми. Кожен проект FST потребує програми з номером 0.

Клік правою кнопкою миші на директорії «Programs» на панелі інструментів проекту відкриває контекстне меню з можливістю вставки нової програми. Або скористайтеся піктограмою «Нова програма» («New Program») чи меню «Вставити» («Insert») (рис. 2.14) і обрати потрібну мову програмування (рис. 2.15). Далі потрібно обрати параметри програми (рис. 2.16).

У цьому прикладі створимо програму у вигляді структурної схеми (Ladder Diagram) (рис. 2.24).

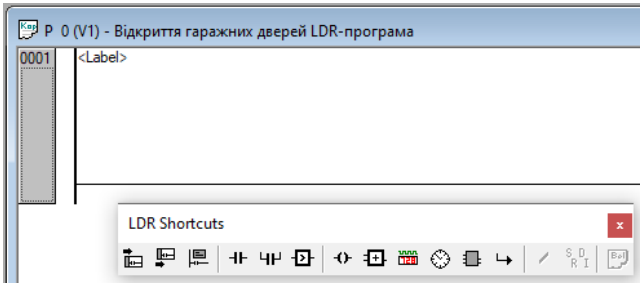


Рисунок 2.24 – LDR-програма керування гаражними воротами

Далі необхідно описати роботу гаражних воріт. Двері повинні відчинятися, якщо:

– кнопка натиснута всередині **АБО**

– кнопка натиснута ззовні **І** одночасно спрацьовує ключ.

У структурній схемі «**АБО**» стає паралельним з'єднанням, «**І**» – послідовним з'єднанням, а «**НЕ**» – розривом (рис. 2.25).

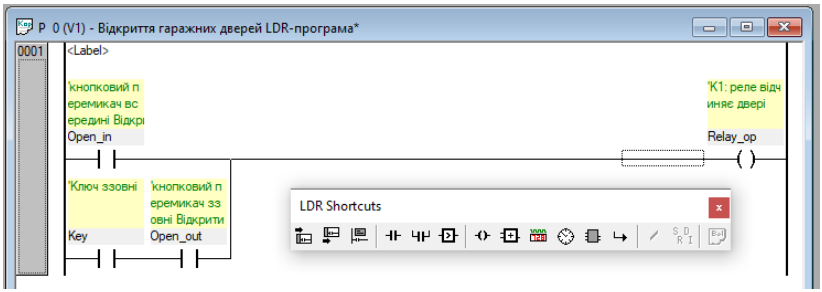


Рисунок 2.25 – LDR-програма керування гаражними воротами

Замість адреси оператора можна використовувати символи, програми на рис. 2.25 і 2.26 є еквівалентними.

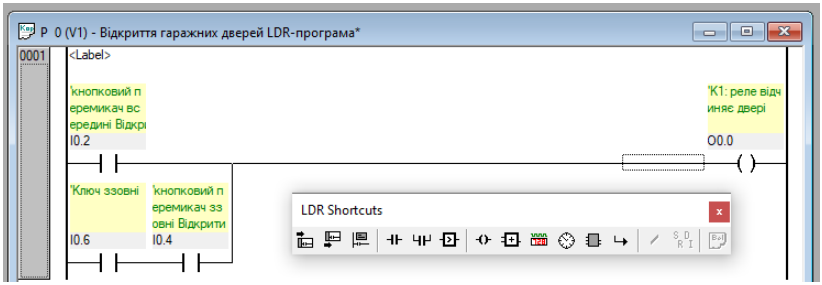


Рисунок 2.26 – LDR-програма керування гаражними воротами

Однак програма працює некоректно: хоча двері починають відчинятися, вони одразу ж зупиняються, якщо одна з кнопок більше не натиснута. Не вистачає «блокування» пам'яті структурної схеми.
Крім того, необхідно визначити, як блокування буде встановлюватися і зніматися. Воно встановлюється за умови, описаної вище, і знімається, як тільки відчиняються гаражні ворота. Програма, що враховує вище зазначене наведена на рис. 2.27.

Зазначимо правила побудови програми у вигляді структурної схеми (Ladder Diagram):

- структурна схема поділяється на мережі. Окрема мережа складається з «контактів» та «дужок»;

– контакти та дужки можуть бути нормально замкненими;

– контакти можна під’єднувати послідовно та паралельно.

Нові мережі вставляються до або після поточної мережі (рис. 2.28).

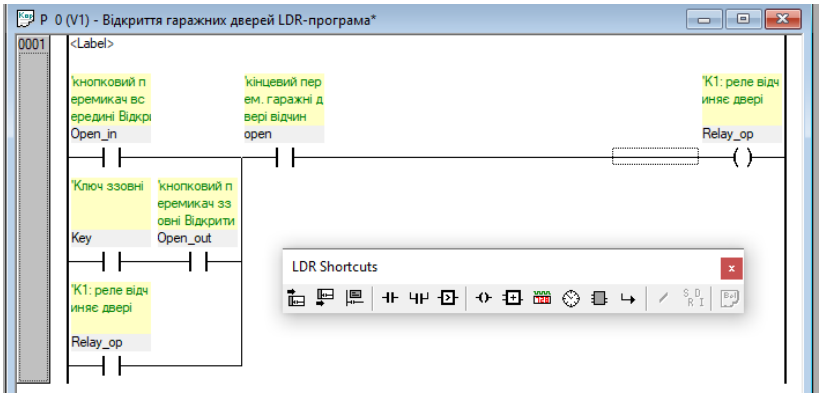


Рисунок 2.27 – LDR-програма керування гаражними воротами з функцією фіксації пам’яті

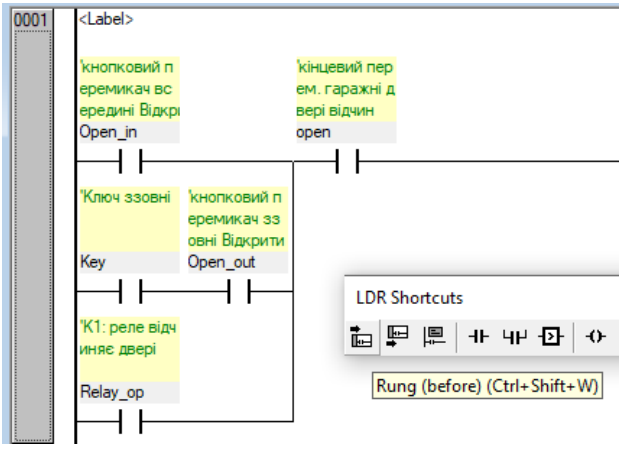


Рисунок 2.28 – Додавання нової мережі до LDR-програми

Додатковий контакт послідовно з існуючим контактом додається шляхом позначення поточного контакту і вставки іншого контакту перед ним за допомогою піктограми «Contact» на панелі «LDR Shortcuts» (рис. 2.29).

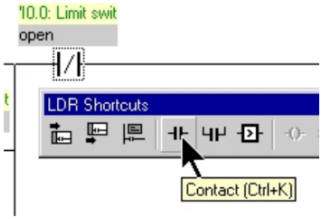


Рисунок 2.29 – Додавання нового послідовно встановленого «контакту» до LDR-програми

Додатковий паралельний контакт додається позначенням контактів, до яких новий контакт повинен бути паралельним, і натисканням піктограми «Parallel contact» на панелі «LDR Shortcuts» (рис. 2.30).

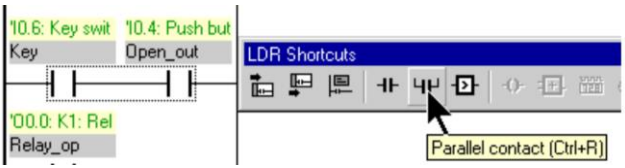


Рисунок 2.30 – Додавання нового паралельно встановленого «контакту» до LDR-програми

Додаткова дужка послідовно з існуючою додається шляхом позначення поточної дужки і вставки іншої за допомогою піктограми «Coil» на панелі «LDR Shortcuts» (рис. 2.31).

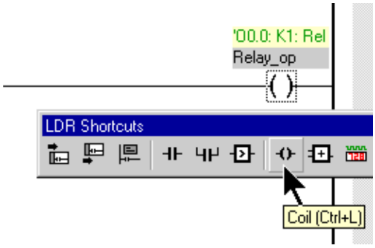


Рисунок 2.31 – Додавання нової послідовно встановленої «дужки» до LDR-програми

Контакт або дужку можна замкнути (або скасувати замикання), позначивши його і замкнувши за допомогою клавіш контекстного меню (рис. 2.32).

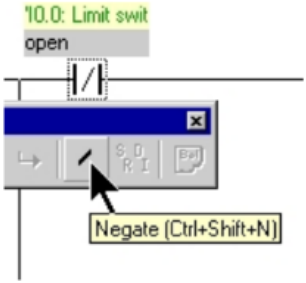


Рисунок 2.32 – Замикання «контакту» у LDR-програмі

Дужці надаються інші властивості системи керування (*S* – встановити; *R* – відновити; *I* – інкремент, відлік вперед; *D* – декремент, відлік назад) шляхом її позначення, а потім прокручування варіантів за допомогою клавіш контекстного меню і натискання потрібної (рис. 2.33).

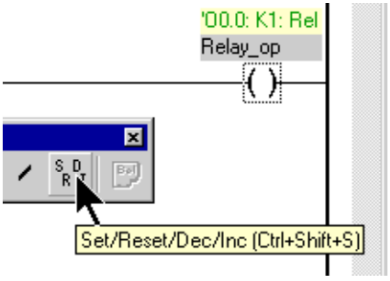


Рисунок 2.33 – Вибір властивостей «дужки» у LDR-програмі

У прикладах на рис. 2.29 – 2.33 коментарі до операторів виводяться у двох рядках. Кількість рядків можна встановити у меню Extras → Preferences... → LDR → Lines for Operand Comments.

При введенні у якості оператора чогось, що FST не розпізнає, вважається, що «Nonsens» є символом нового оператора. Система видає запит, наведений на рис. 2.34. Необхідно натиснути кнопку «Cancel» (або клавішу «Esc») і зробити виправлення.

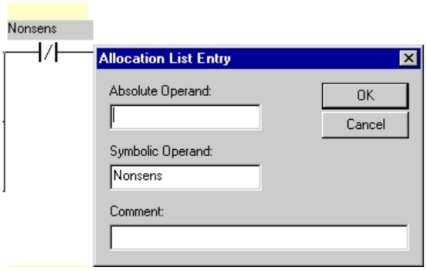


Рисунок 2.34 – Запит у LDR-програмі середовища FST на введення нового оператора

Зважаючи на вище наведене, побудуємо LDR-програму керування роботою гаражних дверей (рис. 2.35).

При закриванні дверей відповідна кнопка повинна залишатися натиснутою безперервно з міркувань безпеки. Отже можна сформулювати це наступним чином: двері зачиняються **якщо** (**IF**) натиснути кнопку CLOSE з внутрішньої **або** (**OR**) зовнішньої сторони і (**AND**) двері не зачинені. **Інакше** (**OTHERWISE**) вони залишаться на місці.

Рисунок 2.35 – Текст LDR-програми, що відповідає безпечному функціонуванню гаражних дверей

Також необхідно додати загальні умови:

– двигун не може працювати в обох напрямках одночасно. Отже, для обох напрямків необхідно задавати одночасно і протилежний напрямок також.

– двигун залишається на місці при одночасному натисканні кнопок CLOSE і OPEN.

З урахуванням вище зазначеного, фінальний текст програми наведений на рис. 2.36. За бажанням до тексту можна додати коментарі.

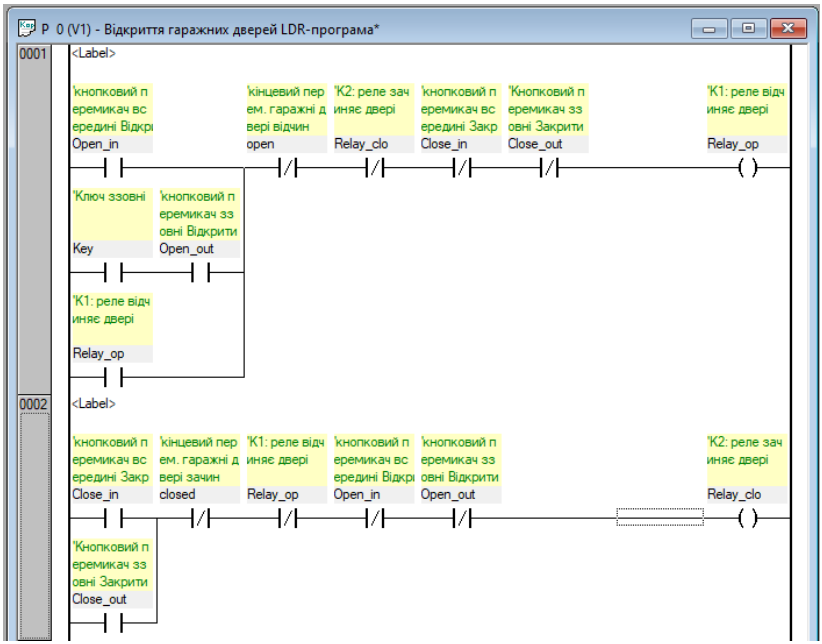


Рисунок 2.36 – LDR-програма керування роботою гаражних дверей

За бажанням до тексту LDR-програми керування роботою гаражних дверей (рис. 2.36) можна додати коментарі.

2.2. Компіляція програми

Під час компіляції виконується синтаксична перевірка, яка шукає у програмі формальні помилки. Клікнувши по іконці «Build Project».

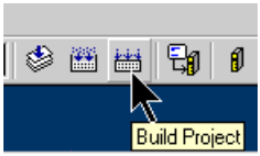


Рисунок 2.37 – Запуск команди синтаксичної перевірки у програмі FST

Програма видасть повідомлення про перевірку та складання списку операторів або структурної схеми (рис. 2.38).

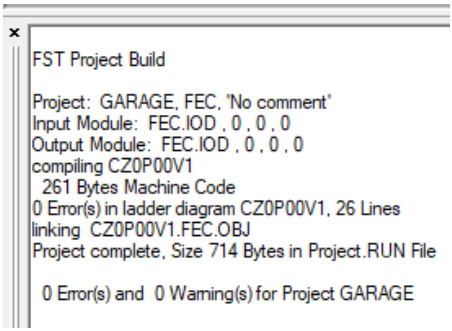


Рисунок 2.38 – Повідомлення про перевірку та складання списку операторів (структурної схеми) у програмі FST

2.2.1 Помилки в списку операторів

У написаній програмі можливо виникнення помилок (рис. 2.39).

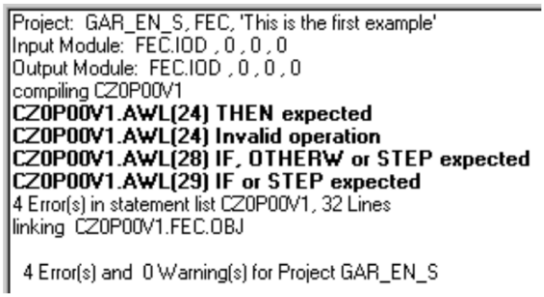


Рисунок 2.39 – Повідомлення про помилки у списку операторів програми FST

У прикладі (рис. 2.39) FST повідомляє про 4 помилки. Ці 4 помилки слід шукати в:

CZ0P00V1.AWL[24] **тоді** (**THEN**) очікується
CZ0P00V1.AWL[24] неправильна операція (**Invalid operation**)
CZ0P00V1.AWL[28] очікується **IF**, **OTHERW** або **STEP**
CZ0P00V1.AWL[29] очікується **IF** або **STEP**

У цьому коді: CZ0P00 – програма (Program) 00 (P00)
V1 – версія 1
[24] – у рядку 24

Двічі клацнувши номер рядка, наприклад [24], FST автоматично переходить до вказаного місця (рис. 2.40).

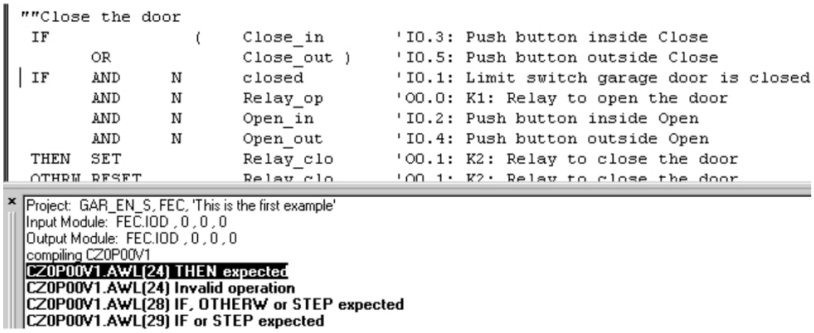


Рисунок 2.40 – Перехід до обраного рядка у програмі FST

Номер поточного рядка також можна побачити у правому нижньому куті рядка стану, якщо курсор знаходиться у вікні програмування. У цьому прикладі у булевій множині двічі введено початок множини **IF**.

2.2.2 Помилки в структурній діаграмі

У прикладі на рис. 2.41 показано помилку.

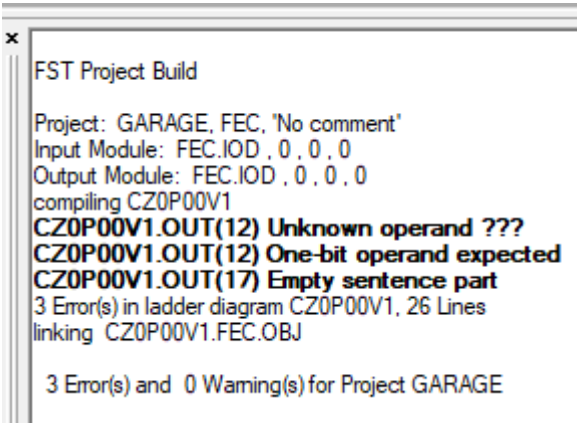


Рисунок 2.41 – Повідомлення про помилки у структурній схемі LDR-програми FST

Три помилки слід шукати в наступних рядках коду:
CZ0P00V1.OUT(12) невідомий оператор ???
CZ0P00V1.OUT(12) очікується однобітний оператор
CZ0P00V1.OUT(17) порожня частина твердження

У цьому коді:

CZ0P00 - Програма (Program) 00 (P00)
V1 – версія 1
[17] – у 17-му рядку скомпільованого коду.

Подвійне клацання на першій помилці відкриває програму 0 і FST автоматично встановлює курсор до вказаного місця (рис. 2.42).

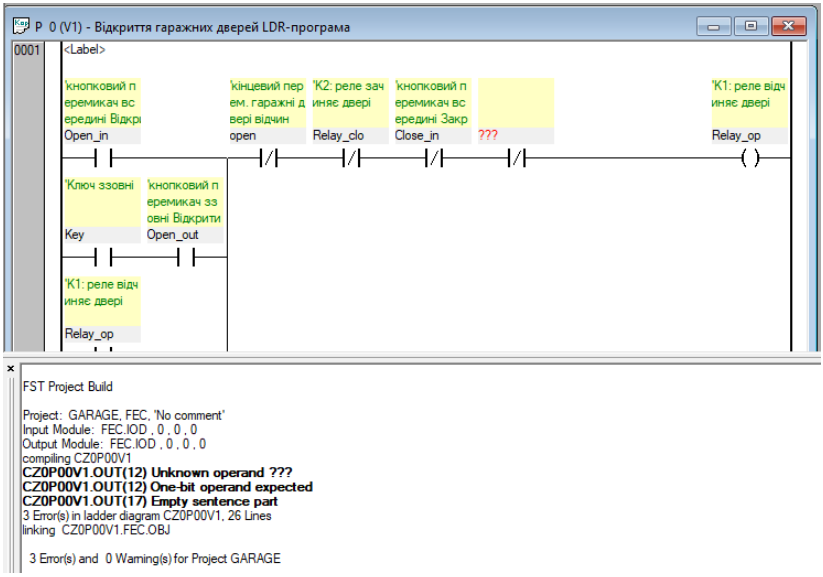


Рисунок 2.42 – Перехід у LDR-програмі FST від повідомлення про помилку до відповідного місця структурної схеми

У прикладі на рис. 2.42 «контакт» введено без зазначення оператора (вхід, вихід, мітка...).

2.3. Завантаження проєкту

• Для підключення контролера необхідно під’єднати його до мережі живлення і за допомогою кабелю для програмування з'єднати з ПК, де міститься написана FST–програма.

У нашому прикладі, FEC Compact FC20, кабель програмування для FEC підключається до порту з позначкою «COM»; до ПК кабель зазвичай підключається до COM1 або COM2.

• Після підключення контролера і підключення кабелю можна перевірити, чи вдалося встановити з'єднання за допомогою іконки «Панель керування» (рис. 2.43).

Рисунок 2.43 – Панель керування FST програми

За умови встановлення з'єднання програма видає повідомлення, наведене на рис. 2.44.

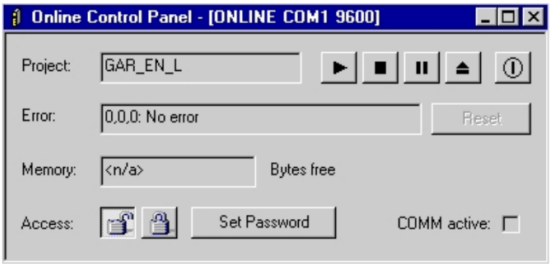


Рисунок 2.44 – Повідомлення про встановлення з’єднання між контролером і ПК

Якщо з'єднання не відбувається, необхідно перевірити:

• Налаштування комп'ютера. Меню «Додатково» («Extras»), пункт «Налаштування...» («Preferences»), вкладка «Зв'язок» («Communication»). Тут можна перевірити, чи обрано правильний номер порта і чи встановлена швидкість передачі даних 9600 (рис. 2.45).

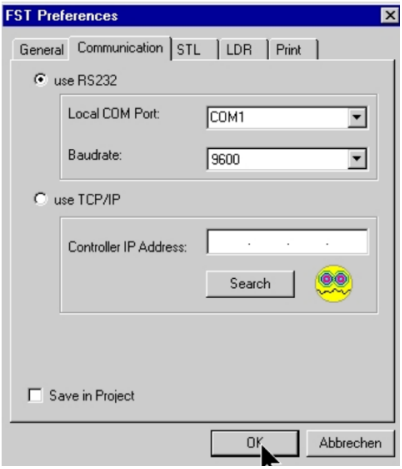


Рисунок 2.45 – Перевірка параметрів з’єднання контролера з ПК

• На контролері повинен горіти зелений світлодіод живлення, а кабель повинен бути підключений до COM-порту. Світлодіод RUN може світитися зеленим, оранжевим або червоним кольором, залежно від того, як він використовувався раніше.

Якщо з'єднання відбулося успішно, ви можете завантажити проект (рис. 2.46).

Рисунок 2.46 – Іконка для завантаження проекту на панелі керування FST програми

У спливаючому вікні повинно з'явитися повідомлення «Завантаження завершено» («Download complete»).

Незалежно від того, як контролер використовувався раніше, необхідно встановити його в режим RUN. Для цього необхідно перевірити індикатор RUN: якщо він зелений, то контролер вже працює в режимі RUN. Якщо світлодіод горить помаранчевим кольором, то необхідно перевести контролер у режим RUN за допомогою повзункового перемикача RUN/STOP (рис. 2.47).

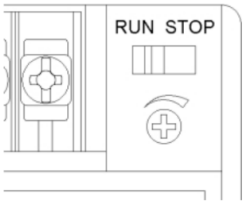


Рисунок 2.47 – Встановлення режиму роботи контролера

Якщо ви використовуєте FEC Standard, для перемикання RUN/STOP використовується поворотний перемикач. Контролер знаходиться в положенні STOP в позиції 0, в усіх інших позиціях – в положенні RUN.

2.4. Перевірка працездатності проєкту

Для перевірки працездатності створеного проєкту автоматизації можна натиснути відповідні кнопки та кінцеві перемикачі і поспостерігати за реакцією гаражних воріт.

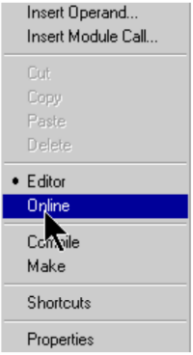


Рисунок 2.48 – Контекстне меню для перевірки роботи програми

Однак також важливо перевірити роботу контролера і відповідність реакцій у проєкті автоматизації на зовнішні збурення. Для цього є дві основні можливості: по-перше, можна спостерігати за програмою, а по-друге – за входами і виходами.

– Щоб спостерігати за роботою програми відкрийте Програму 0 (Program 0) і клікніть правою кнопкою миші у цьому вікні.

Відкриється контекстне меню, незалежно від того, чи написана програма у вигляді списку операторів, чи у вигляді структурної схеми, яке міститься в меню у вкладці «Онлайн» (рис. 2.48).

Увімкніть онлайн-режим, тоді у вікні програми буде показано стан окремих змінних (STL) (рис. 2.49) або чи виконано умову в межах рядка (LDR) (рис. 2.50).

```
""Open the door
 IF                  Open_in          OFF  'IO.2: Push button inside Open
       OR       (    Open_out         OFF  'IO.4: Push button outside Open
       AND           Key       )      OFF  'IO.6: Key switch outside
       AND      N    Relay_clo        OFF  'OO.1: K2: Relay to close the door
       AND      N    Close_in         OFF  'IO.3: Push button inside Close
       AND      N    Close_out        OFF  'IO.5: Push button outside Close
       AND      N    open             OFF  'IO.0: Limit switch garage door is open
 THEN  SET          Relay_op         OFF  'OO.0: K1: Relay to open the door

""Stop the door
 IF                  open             OFF  'IO.0: Limit switch garage door is open
       OR           Close_in         OFF  'IO.3: Push button inside Close
       OR           Close_out        OFF  'IO.5: Push button outside Close
 THEN  RESET        Relay_op         OFF  'OO.0: K1: Relay to open the door

""Close the door
 IF             (    Close_in         OFF  'IO.3: Push button inside Close
       OR           Close_out )      OFF  'IO.5: Push button outside Close
       AND      N    closed           OFF  'IO.1: Limit switch garage door is closed
```

Рисунок 2.49 – Вікно перевірки стану змінних для STL-програми у FST

У вікні програми можна спостерігати за складовими в онлайн-режимі в межах видимого фрагмента програми.

Однак часто необхідно спостерігати за змінними незалежно від того, яка частина програми відображена у вікні. Тоді можна викликати команду «Онлайн-відображення» («Online Display») (рис. 2.51).

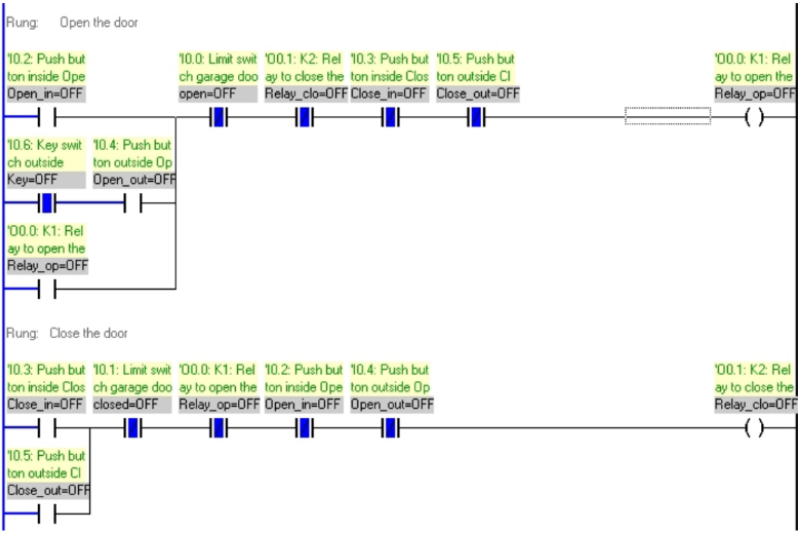


Рисунок 2.50 – Вікно перевірки стану змінних для LDR-програми у FST

Рисунок 2.51 – Виклик команди «Online Display» на панелі керування FST програми

На дисплеї (рис. 2.52) видно, що входи I0.3, I0.4 та I0.7 активовані. Такі можливості спостереження полегшують введення проєкту в експлуатацію.

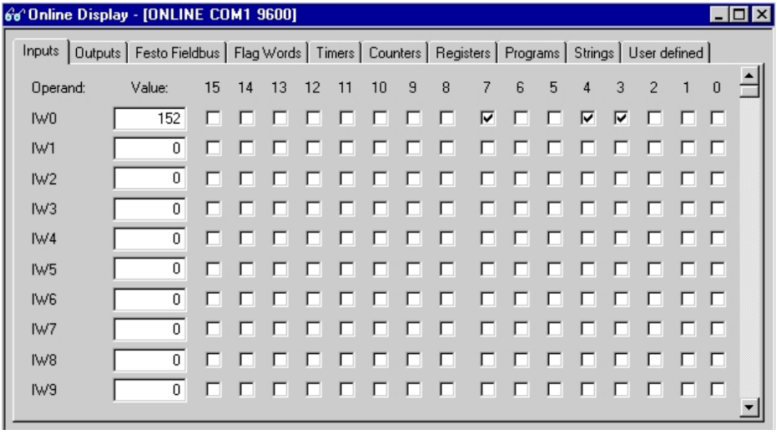


Рисунок 2.52 – Результат використання команди «Online Display» у FST програмі

2.5. Проєктна документація

FST дозволяє автоматично створювати документацію до проекту:
– коментарі в програмі;
– проєктну документацію (рис. 2.53).

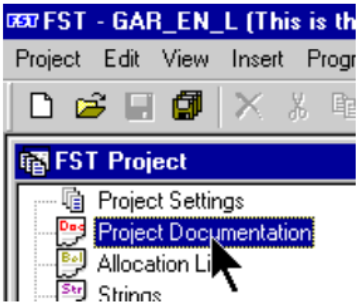


Рисунок 2.53 – Вибір вкладки «Проєктна документація» («Project Documentation») у вікні проєкту FST програми

Зазвичай документація проєкту є пустим текстовим файлом. Якщо двічі клікнути по вкладці «Project Documentation», відкриється файл project.txt у текстовому редакторі, активованому у системі Windows (рис. 2.54). Файл зберігається в каталозі поточного проєкту.

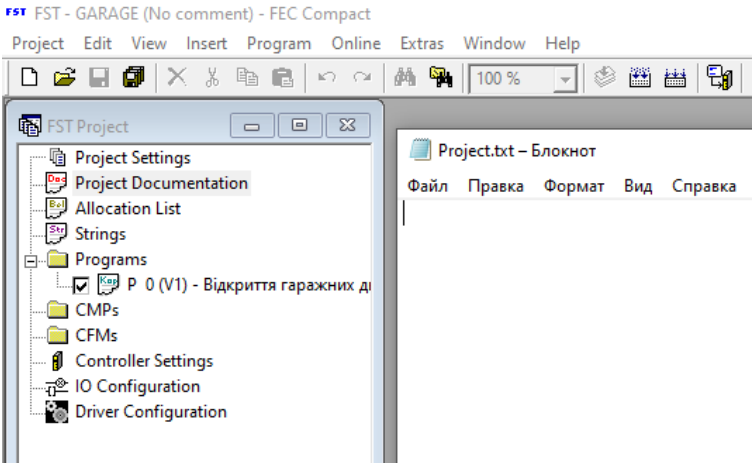


Рисунок 2.54 – Відкритий файл project.txt в результаті використання команди «Project Documentation»

Створений проект можна роздрукувати (рис. 2.55).

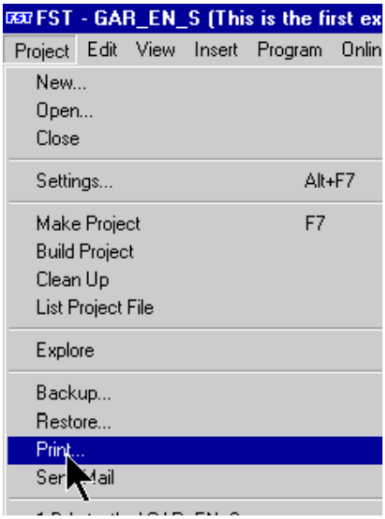


Рисунок 2.55 – Вкладка «Друк» («Print») у FST програмі

Причому можна обрати окремі компоненти проектної документації, які необхідно надрукувати (рис. 2.56).

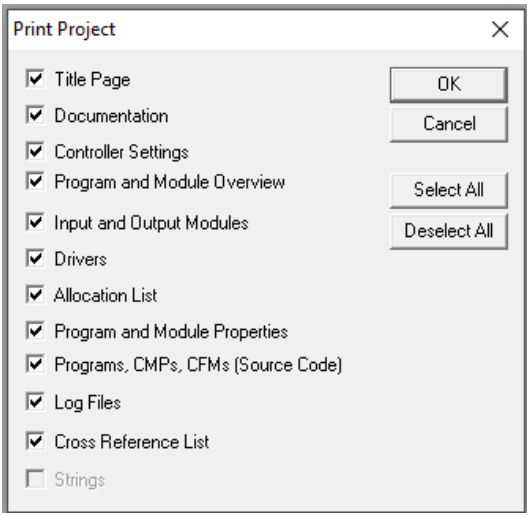


Рисунок 2.56 – Вибір компонентів проєкту, які необхідно роздрукувати

2.6. Проєкт «Гараж» («Garage»)

Зразки виконаних проєктів містяться у каталозі проєктів FST. Для проекту «Garage»:

– Gar_en_s (запрограмовано у вигляді списку операторів)

– Gar_en_l (запрограмовано у вигляді структурної схеми).

2.7 Висновки

При написанні FST-програми робота розпочинається із створення проєкту. Проєкт необхідний для кожного окремого контролера (кожного процесора). Порядок дій відображений у таблиці 2.1.

Таблиця 2.1 – Базові процедури програмування для середовища FST

№ п.п.	Задача	Коментар
1	Створити новий проєкт	У меню Project
2	Назвати проєкт	У діалоговому вікні
3	Обрати відповідний контролер, ввести коментар до проєкту	Можна використовувати програмне забезпечення FST для програмування різних процесорів сімейства контролерів
4	Конфігурація 1/0 (вводу/виводу)	Кожен проєкт автоматизації має вхідні та вихідних дані.
5	Програмування	Наявність базової програми (Program 0) є обов'язковою. Прийняття рішення про використання структурної схеми чи списку операторів
6	Створити проєкт	Система пропонує автоматично
7	Завантажити проєкт	Для цього має бути встановлено з'єднання з контролером
8	Перевірка	Надає можливість виправити, покращити, оптимізувати програму
9	Документ	Опис, коментарі, можливість роздрукувати

– В процесі написання програми, як правило, спочатку створюється список розподілу (Allocation list) для входів і виходів.

– Проекти FST завжди потребують Програми 0.

– Список операторів

– Програмування здійснюється за допомогою операторів **IF ... THEN ... OTHRW** всередині програми.

– **IF** і **THEN** повинні використовуватися один раз у кожній булевій множині.

– **OTHRW** може використовуватися, але не обов'язково.

– Структурна схема

– Мережі програмуються всередині програми.

– Мережа складається з «контактів» і «дужок».

ДОДАТОК 1
ІНДИВІДУАЛЬНЕ ЗАВДАННЯ:
КЕРУВАННЯ СТРІЧКОЮ КОНВЕЄРА

Мета роботи – набуття практичних навичок у створенні пневмоелектричних схем та схем з мікроконтролерним керуванням; ознайомлення з загальними принципами складання і моделювання роботи схем пневмоприводів з електричним/мікроконтролерним керуванням з використанням програми FluidSim.

1.1 Завдання:

– розробити принципові пнемо- і електросхеми з прямим керуванням зворотно-поступальним рухом штока пневмоциліндра;

– створити FST-програму мікороконтролерного керування рухом штока пневмоциліндра;

– перевірити працездатність розроблених схем і програми у програмі FluidSim;

– зібрати розроблені схему на експериментальному стенді.

1.2 Постановка задачі

За допомогою конвеєрної стрічки деталі переміщуються через рівні проміжки часу до робочих місць, які розташовані послідовно одне за одним.

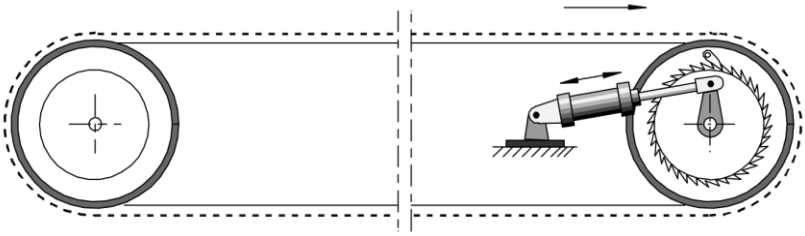


Рисунок 1.1 – Схема керування стрічкою конвеєра (зворотно-поступальним рухом пневмоциліндра)

При натисканні кнопкового перемикача з фіксацією ведуче колесо починає крокову подачу деталей за рахунок зворотно-поступального переміщення штока циліндра, поєднаного з колесом за

допомогою кривошипа (рис. 1.1). При повторному натисканні кнопкового перемикача рух припиняється.

1.3 Побудова принципових пнемо- та електросхеми

Для побудови пнемо- та електросхеми системи керування стрічкою конвеєра (рис. 1.1) використаємо наступні елементи:

Кількість	Найменування
1	пневмоциліндр двосторонньої дії
1	блок підготовки повітря з 3/2 розподільником
1	колектор
1	5/2 розподільник з двостороннім електромагнітним керуванням
1	блок живлення
1	панель вводу сигналів, електрична
1	електричний кінцевий перемикач з ввімкненням зліва
2	електричний кінцевий перемикач з ввімкненням зправа

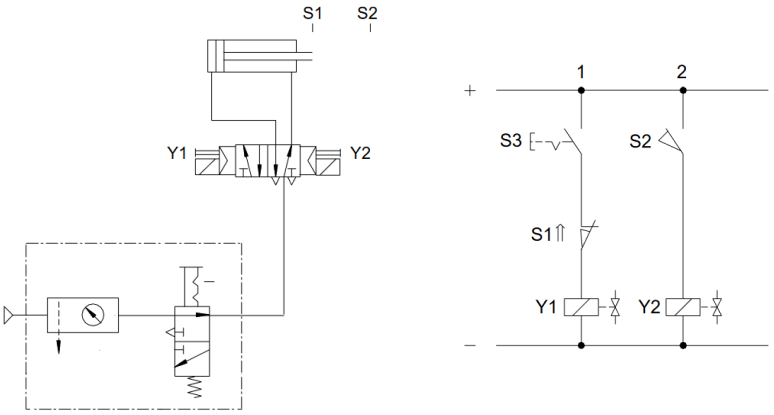


Рисунок 1.2 – Монтажна пневмо- та електросхема керування зворотно-поступальним рухом штока пневмоциліндра

При натисканні кнопкового перемикача з фіксацією S3 замикається електричний ланцюг подачі струму на електромагніт Y1 і переключається 5/2–розподільник з двостороннім керуванням (рис. 1.2). Шток циліндра двосторонньої дії висувається і в передньому крайньому положенні натискає кінцевий перемикач S2. З початком руху штока кінцевий перемикач S1 відключається і розмикає ланцюг з електромагнітом Y1.

При спрацюванні вимикача S2 замикається електричний ланцюг електромагніта Y2 і 5/2–розподільник повертається у вихідне

положення. Шток циліндра втягується і у вихідному положенні знову натискає на кінцевий перемикач S1. Через те, що з початком зворотного руху штока кінцевий перемикач S2 вимикається, розмикаючи ланцюг з електромагнітом Y2, по приходу штока у вихідне положення, завдяки затисненій кнопці S3 і спрацюванню S1, шток знову починає висуватися. Отже організовується робочий цикл.

1.4 Перевірка працездатність розроблених схем у програмі FluidSim

За схемами на рис. 1.2 у програмі FluidSim збираємо пневмо-електросхему керування зворотно-поступальним рухом штока пневмоциліндра (рис. 1.3).

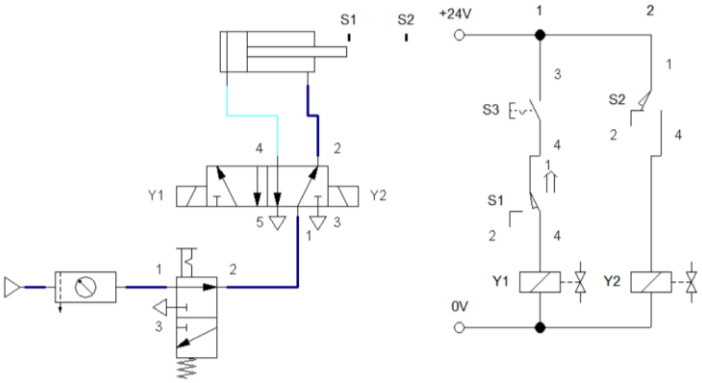


Рисунок 1.3 – Реалізація пнемо- електросхеми керування зворотно-поступальним рухом штока пневмоциліндра у програмі FluidSim

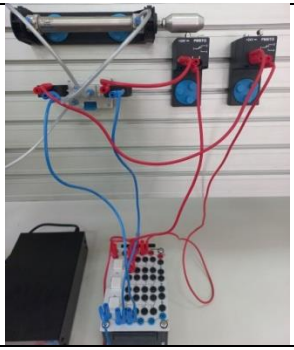


Рисунок 1.4 – Змонтована схема керування зворотно-поступальним рухом штока пневмоциліндра на експериментальному стенді

Після того, як ми впевнилися у працездатності розробленої схеми за допомогою програми FluidSim, переходимо до її монтажу на експериментальному стенді (рис. 1.4).

1.5 Створення автоматичної системи керування зворотно-поступальним рухом штока пневмоциліндра на основі мікроконтролерного керування

На рис. 1.5 наведено бістабільний пневмоциліндр (1) з контролем положення штока за допомогою кінцевих перемикачів (3). Зміна напрямку руху штока забезпечується за допомогою 2/4-

пнеморозподільника (2) з електрокеруванням. Схеми на рис. 1.4 і 1.5 ідентичні.

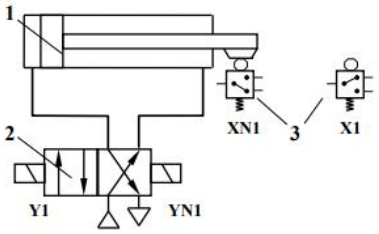


Рисунок 1.5 – Функціональна схема бістабільного пневмоциліндра з мікроконтролерним керуванням

Алгоритм функціонування технологічної системи, що складається з бістабільного пневмоциліндра з контролем положення штока за допомогою кінцевих перемикачів і 2-позиційного 4-лінійного пневморозподільника з електрокеруванням буде наступним:

IF <УМОВА ОСНОВНОЇ КОМАНДИ>
THEN RESET YNn
SET Yn
IF <УМОВА ЗВОРОТНОЇ КОМАНДИ>
THEN RESET Yn
SET YNn

Створимо FST-програму керування системою, наведеною на рис. 1.5 (за алгоритмом, наведеним вище) – 2_CILIND.

Спершу, керуючись алгоритмом і функціональною схемою на рис. 1.5, заповнимо список операторів (Allocation List) (рис. 1.6).

Із рис. 1.6 видно, що програма має 4 входи (I0.0…O0.3) і 2 виходи (O0.0…O0.1) і враховую обидва стани кнопкового та кінцевих перемикачів (наявний/відсутній сигнал).

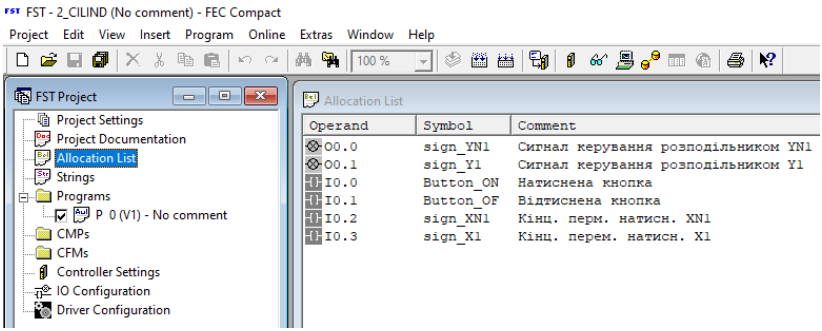


Рисунок 1.6 – Список операторів програми 2_CILIND із внесеними даними

```
                                    " Extension the road of cilinder
STEP Extension

 IF                  Button_ON      'Натиснена кнопка
       AND           sign_XN1       'Кінц. перм. натисн. XN1
 THEN  SET           sign_Y1        'Сигнал керування розподільником Y1
       RESET         sign_YN1       'Сигнал керування розподільником YN1

                                    " Retract the road of cilinder
STEP Retractio
 IF                  sign_X1        'Кінц. перем. натисн. X1
 THEN  SET           sign_YN1       'Сигнал керування розподільником YN1
       RESET         sign_Y1        'Сигнал керування розподільником Y1
       JMP TO Extension

 IF                  Button_OF      'Відтиснена кнопка
       AND           sign_XN1       'Кінц. перм. натисн. XN1
 THEN  RESET         sign_Y1        'Сигнал керування розподільником Y1
       SET           sign_YN1       'Сигнал керування розподільником YN1

       JMP TO Extension
 IF                  Button_OF      'Відтиснена кнопка
       AND           sign_X1        'Кінц. перем. натисн. X1
 THEN  SET           sign_YN1       'Сигнал керування розподільником YN1
       RESET         sign_Y1        'Сигнал керування розподільником Y1

       JMP TO Extension
 IF                  Button_OF      'Відтиснена кнопка
       AND     N     sign_X1        'Кінц. перем. натисн. X1
       AND     N     sign_XN1       'Кінц. перм. натисн. XN1
 THEN  SET           sign_YN1       'Сигнал керування розподільником YN1
       RESET         sign_Y1        'Сигнал керування розподільником Y1

       JMP TO Extension
```

Рисунок 1.7 – Текст FST-програми 2_CILIND

1.6 Монтаж на лабораторному стенді автоматичної системи керування зворотно-поступальним рухом штока пневмоциліндра на основі мікроконтролерного керування

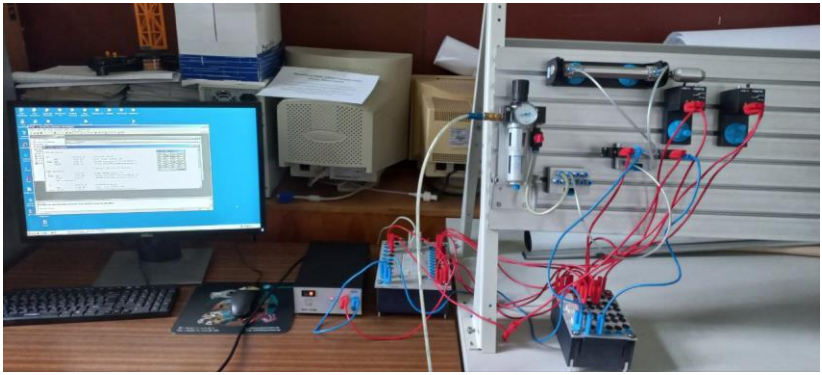


Рисунок 1.8 – Змонтована на лабораторному стенді автоматична система керування зворотно-поступальним рухом пневмоциліндра

1.7. Програма керування зворотно-поступальним рухом штока пневмоциліндра у вигляді структурної схеми

Технічною системою керування стрічкою конвеєра, наведеною на рис. 1.1, можна керувати також за допомогою LDR-програми, створеної у середовищі FST (рис. 1.9).

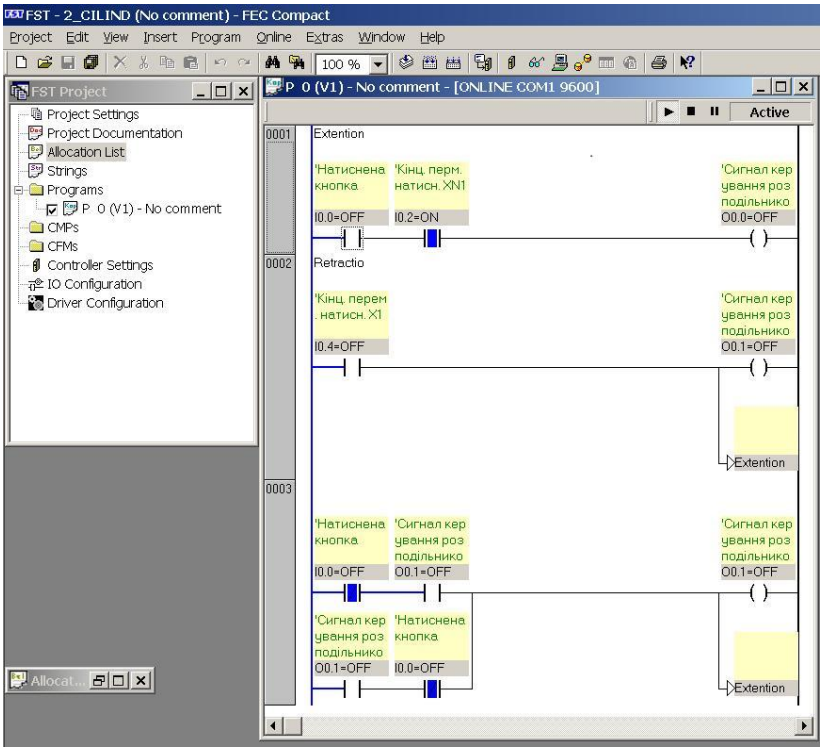


Рисунок 1.9 – LDR-програма керування зворотно-поступальним рухом пневмоциліндра, створена у середовищі FST

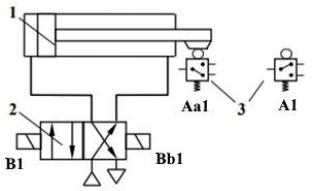
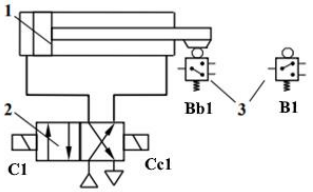
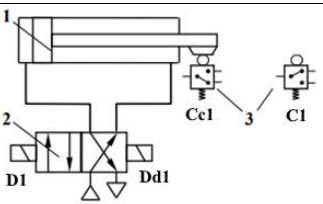
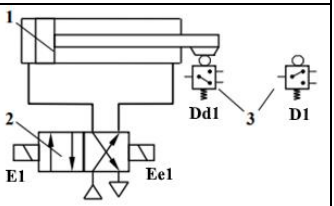
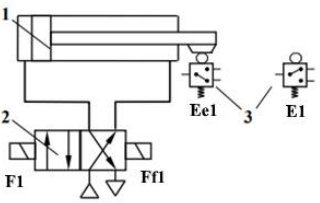
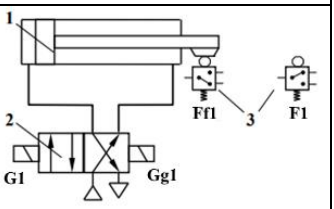
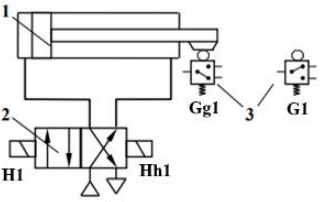
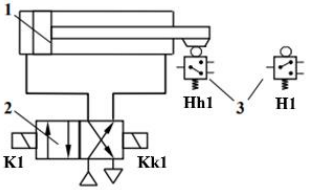
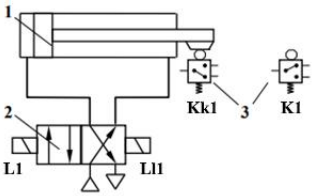
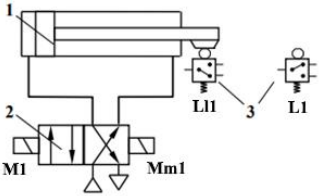
Як видно з рис. 1.9, програма у вигляді структурної схеми є спрощеною, порівняно операторною формою (рис. 1.7) і потребує лише трьох входів (I0.0, I0.2 та I0.4) і двох виходів (O0.0 та O0.1). При цьому негативний стан умови зафарбований синім кольором.

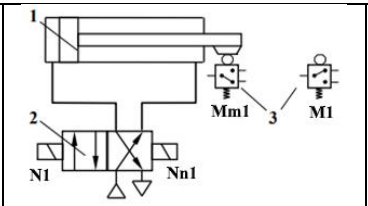
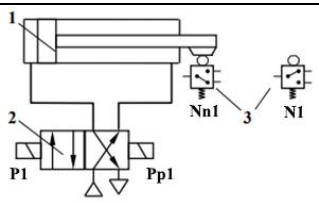
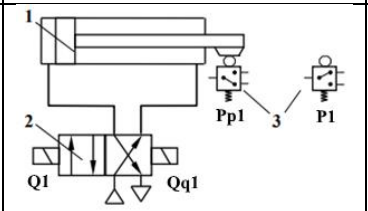
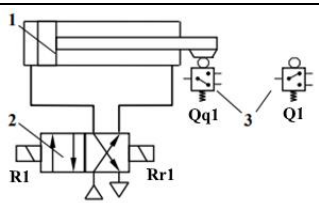
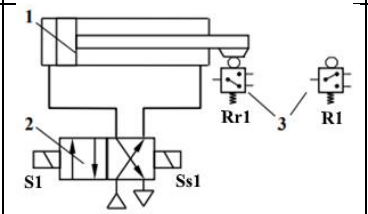
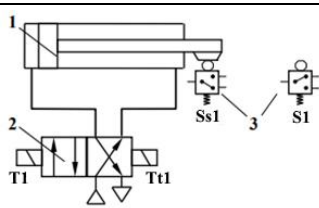
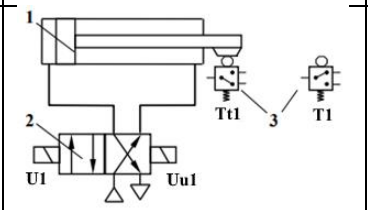
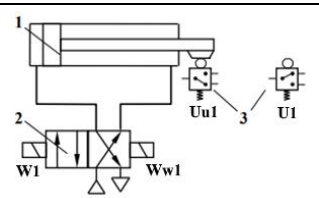
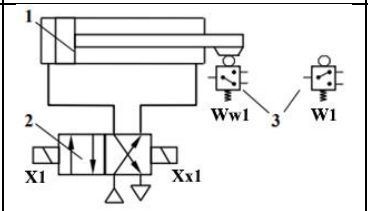
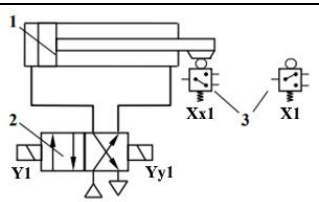
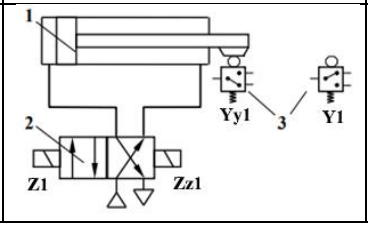
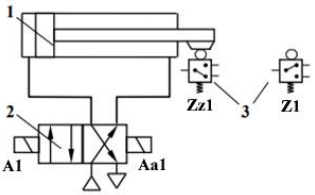
1.8. Завдання для самостійного виконання

Розробити автоматичну систему керування зворотно-поступальним рухом штока двостороннього пневмоциліндра на основі мікроконтролерного керування за наданим у таблиці 1.1 варіантом схеми (варіант відповідає номеру у списку групи). Програму керування навести у двох варіантах: у вигляді списку операторів та структурної схеми.

Звіт до завдання повинен містити короткі теоретичні відомості, конструктивну та функціональну схеми пристрою, текст програми (відповідно до варіанту у таблиці 1.1), висновок.

Таблиця 1.1 – Варіанти лабораторної роботи № 1 для самостійного виконання

№ вар.	Функціональна схема	№ вар.	Функціональна схема
1		2	
3		4	
5		6	
7		8	
9		10	

11		12	
13		14	
15		16	
17		18	
19		20	
21		22	

Список літератури

1. Проєктування та випробування пневматичних керуючих систем. Методичні вказівки до виконання лабораторних та практичних робіт з навчальних дисциплін «Сучасні елементи гідропневмосистем», «Основи теорії пневмоприводу», «Пневматичне і вакуумне обладнання гідропневмосистем» для студентів денної та заочної форми навчання за спеціальністю «Прикладна механіка» / уклад.: Г. А. Крутіков, М. Г. Стрижак, П. Я. Ніконов; Харків: НТУ "ХПІ", 2024. – 69 с. https://repository.kpi.kharkov.ua/handle/KhPI-Press/73821.

2. Проєктування та випробування електропневматичних керуючих систем. Методичні вказівки до виконання лабораторних та практичних робіт з навчальної дисципліни «Електрогідравлічні й електропневматичні перетворювачі гідро пневмосистем» для студентів ден. та заочн. форми навчання за спеціальністю «Прикладна механіка» / уклад.: В. В. Клітной, М. Г. Стрижак, П. Я. Ніконов; Харків: НТУ "ХПІ", 2024. – 64 с. – URI: https://repository.kpi.kharkov.ua/handle/KhPI-Press/79772.

3. Функціональні модулі систем мехатроніки з пневматичними, електромеханічними та гідравлічними виконавчими пристроями [Електронний ресурс]:навч. Посіб / О.П. Губарев, О.С. Ганпанцурова, К.О. Бєліков, А.М. Муращенко; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2019. – 104 с.

4. Festo. Air Box Type GHDA-FQ-M-FDMJ-A—Operating Instructions; Festo: Esslingen, Germany, 2006.

5. Management training course – PNEUMATICS – TP101. P. Crosser, F. Ebel. Festo. 2006. Kyiv.

6. Management training course – PNEUMATICS – TP201. P. Crosser, F. Ebel. Festo. 2006. Kyiv.

7. ISO 1219-1:2014 – Graphic symbols and principal schemes. National standard of Ukraine. 2017. Kyiv.

ЗМІСТ

Навчальне видання

Методичні вказівки до виконання
індивідуального завдання
з навчальної дисципліни
«Програмування автоматизованих технічних комплексів»
для студентів денної та заочної форми навчання за спеціальністю
«Прикладна механіка», освітня програма «Моделювання технічних
систем»

Укладачі:
КЛІТНОЙ Володимир Вікторович
СТРИЖАК Мар'яна Георгіївна

Відповідальний за випуск проф. Заковоротний О. Ю.
Роботу до видання рекомендував доц. Шевцов В. М.

В авторській редакції

План 2025 р., поз. 655
Підп. до друку Формат 60×84 1/16.
Папір офсет. Друк ризографічний. Ум. друк. арк. 3.
Обл. вид. арк. Наклад 50 прим. Замовлення №

Видавничий центр НТУ «ХПІ»,
вул. Кирпичова, 2, м. Харків, 61002
Свідоцтво суб'єкта видавничої справи ДК № 5478 від 21.08.2017 р.

Електронна версія